Reflection for COMPSCI 4ZP6

RatBAT: Rat Behavioral Analysis Tool

Team 8

Brandon Carrasco

Daniel Locke

Jamie Wong

Inoday Yadav

GitHub repo link: https://github.com/brandonc-edu/RatBAT

Achievements:

We successfully developed a web-based platform that enables users to filter, access, process, and analyze behavioral data from the Szechtman Lab Collection on the FRDR repository. We built a custom MariaDB relational database to store time series data and metadata variables. We built a preprocessing suite, which can smooth and label raw time series data. Our smoothing algorithm utilizes custom implementations of **Locally Weighted Scatterplot Smoothing** (LOWESS), and **Repeated Running Medians** smoothing (RRM) in order to provide robust smoothing. The segmentation algorithm segments rodent movement into periods of lingering (low activity) and progression (high activity) by modeling the rodent's approximate velocity as a **Gaussian Mixture Model (GMM)** using **Expectation Maximization (EM)** and uses chi-squared hypothesis testing to determine optimal number of gaussians in the GMM. All components of these preprocessed algorithms were implemented by us based on *Journal of Neuroscience Methods 96 (2000) 119-131* and *Journal of Neuroscience Methods 133 (2004) 161-172*. We constructed an extensible summary measure calculation interface that makes it easy to create new summary measures via generic method calling, summary measure method templates, and inter-summary measure dependency handling. Additionally, we implemented several summary measure algorithms while also optimizing the interface to prevent recalculation of intermediary data shared across summary measures, ensuring that the average calculation time hovered around one second for each summary measure. Finally, we have made our summary measures algorithms and interface open-source and published it as a package for available users to use in their own local projects.

P0 and P1 original requirements and the percentage of completion

Total: 17, completed: 13 (76%)
Version 0 uncompleted:

P0: The system must allow users to choose which raw data types to download.

- Due to the limited hard drive and RAM capacity on our long term server, we could only store time series data and not video or path plot data (there is ~10TB of video data in the library, and we have a hard drive capacity of 1TB). Instead, we allow users to download direct download links to videos and path plots so they can download at their own discretion.

P1: The system must display any raw or pre-processed time series selected by the user

- This requirement was removed due to performance concerns and limited added value. Displaying 98000+ rows of data for each selected trial would have introduced significant rendering and UI complexity without supporting the main goals of the web platform.

P0: The system must allow the user to submit Globus credentials to permit bulk download from FRDR.

- We switched from Globus to HTTP for file transfer as the required Globus functionality needed an organizational subscription, which McMaster does not have.

P1: The system must store any preprocessed time series data and summary measures computed on the web platform within a database.

- Due to the very limited hard drive and RAM capacity on our long-term server, storing multiple versions of preprocessed data with different parameters used for

preprocessing was infeasible. Thus, our database only stores preprocessed data, which was processed using our default parameters.

What worked:

Consistent communication was essential to our team's progress and cohesion. We held two weekly meetings where each member had the opportunity to update other team members on their progress, receive feedback, and collaboratively troubleshoot issues. Holding regular meetings allowed us to stay aligned and maintain a sense of collaboration and teamwork on the project. Additionally, our monthly meetings with our supervisors also provided valuable guidance and feedback. These sessions helped in keeping us accountable to our stakeholders, ensuring we met key objectives and kept the direction of the project aligned with the expectations of our supervisors.

We dedicated a significant amount of time to understanding the problem domain related to our project to fully grasp the context in which our application would be used. This foundational knowledge enabled us to develop an app that directly addressed the needs of our stakeholders, ensuring our solution was practical and relevant for those working in the domain.

What did not work:

We searched extensively for web hosting options for our supervisors, but were not able to find a cost-effective option that met their needs in the beginning. Had we started with the CAS server earlier we could have had a better estimate for our compute resources requirements and saved time.

Although two weekly meetings helped maintain progress, our lack of flexibility became a challenge as the workload increased, particularly near the end of the course. This led to missing meetings, delayed feedback, and added pressure around deadlines. Greater adaptability in our meeting schedule or more proactive planning around peak academic periods could have helped reduce this strain.

Another challenge we faced was the timing of integrating the end-to-end functionality of the web platform. By isolating components early on, we postponed connecting the full app until late in development, which led to a last-minute rush for the final video demo. Prioritizing integration earlier would have allowed for earlier testing and more meaningful feedback throughout the process.

What we learned:

We learned how to work efficiently as a team and effectively communicate our ideas, concerns, and questions to each other. We learned how to pace ourselves and stay on track through a 6 month long project using GitHub Issues for planning weekly tasks. We learned how to work closely with stakeholders to design and revise a product based on requirements and rounds of feedback.

Based on our challenges, we also learned the importance of early integration, consistent communication, and understanding the user's domain. Regular feedback helped shape our direction, while delays in connecting components taught us to prioritize early testing. Furthermore, we learned that flexibility in planning is crucial, especially when balancing heavy workloads.